(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 358 292 B1

(12) EUROPEAN PATENT SPECIFICATION

(45) Date of publication and mention of the grant of the patent:
10.09.1997 Bulletin 1997/37

(51) Int Cl.6: **G06F 9/445**, G06F 15/16, G06F 9/44

(21) Application number: **89302132.9**

(22) Date of filing: **03.03.1989**

(54) Remote boot

Fern-Urlader

Chargement initial à distance

(84) Designated Contracting States:
DE FR GB NL

(30) Priority: **06.09.1988 US 240955**

(43) Date of publication of application:
14.03.1990 Bulletin 1990/11

(73) Proprietor: **DIGITAL EQUIPMENT CORPORATION**
Maynard Massachusetts 01754-1418 (US)

(72) Inventor: **Flaherty, James E.**
Hudson Massachusetts 01749 (US)

(74) Representative: **Goodman, Christopher et al**
Eric Potter & Clarkson
St. Mary's Court
St. Mary's Gate
Nottingham NG1 1LE (GB)

(56) References cited:
- IBM TECHNICAL DISCLOSURE BULLETIN, vol. 26, no. 10B, March 1984, page 5609, New York, US; P.E. MILLING: "Remote IPL of diskette-driven computers"
- IBM TECHNICAL DISCLOSURE BULLETIN, vol. 13, no. 5, October 1970, page 1203, New York, US; B.B. YOUNG et al.: "Remote initial program lead and library maintenance for satellite computers"
- "The MS-DOS Encyclopedia", 1988, pages 61-82, article 2, Microsoft press, Redmond, Washington, US
- IBM TECHNICAL DISCLOSURE BULLETIN, vol. 27, no. 1B, June 1984, pages 481-482, New York, US; D.J. BRADLEY et al.: "feature extensions for IBM personal computer

Printed by Jouve, 75001 PARIS (FR)

Description

Background

A computer includes both a physical machine, namely the hardware, and the instructions which cause the physical machine to operate, namely the software. Software includes both application and system programs. If the program is simply to do tasks for a user, such as solving specific problems, it is referred to as application software. If a program controls the hardware of the computer and the execution of the application programs, it is called system software. System software further includes the operating system, the program which controls the actual computer or central processing unit (CPU), and device drivers which control the input and output devices (I/O) such as printers and terminals.

A general purpose computer is fairly complicated. Usually there will be a queue of application programs waiting to use the central processing unit. The operating system will need to determine which program will run next, how much of the CPU time it will be allowed to use and what other computer resources the application will be allowed to use. Further, each application program will require a special input or output device and the application program must transfer its data to the operating system which controls the device drivers.

The operating system is generally too large and complex to be stored in non-volatile Read Only Memory (ROM). Additionally, there are generally periodic changes made to the operating system which makes it impractical to store it in ROM. The operating system is usually stored on a magnetic disk and read into Random Access Memory (RAM) to be executed. The problem is that the access to the disk requires the disk hardware driver and the rest of the system software. It would therefore appear that there is no mechanism to start the computer, since to read the operating system off the disk requires that the operating system be present.

To get around this problem a technique called bootstrapping or booting is used. In booting, a small program is provided in a special ROM, called the boot-ROM. When the computer is first started, the contents in the boot-ROM are read and that program executed. The boot-ROM program is a small program which instructs the CPU where on a disk to find a larger boot program, which is termed the "boot-block" program. It also instructs the CPU how to load the program into memory and execute it. The boot-block program is executed to copy the operating system off the disk. Once the operating system is in memory, it is executed and the computer is completely functional.

A network of general purpose computers may be constructed by having a number of general purpose computers, termed nodes, communicate with one another over a communications link. If the computers are distant from each other, the network is termed a wide area network (WAN), while, if they are close together, for example, in the same building, it is called a local area network (LAN). In one type of LAN, each computer or node is connected to a common communication link by way of an interface, called a network device.

The computers in a network communicate by sending messages to each other. Each message consists of a header field, which contains an identifier which comprises the address of the network device of the computer to which the message is directed and an identifier which comprises the address of the network device of the computer sending the message, and a data field, which contains the information being sent between the computers.

Each network device monitors the messages on the link and copies the message from the link into the computer's memory and notifies the computer if the message header contains the network device's address or if a message header includes a broadcast address, indicating that the message is being sent to all devices on the network or a multicast address, indicating that the message is being sent to all devices within a certain address range. When a computer wishes to transmit information to another computer, it attaches the address of the intended recipient to its own address and attaches both to the information to form a message, which is transmitted over the communication link.

The availability of local area networks has added to the versatility of computers. It did not take users long to realize that the network could make files on disks belonging to one computer system on the network available to all computers on the network. Programs were developed which simplified access to the files on the disks of another computer system. Eventually the concept evolved to assign special functions to certain computers on the network. For example, one computer would assign logical names to each physical device accessible to the network. In that way a user instead of requesting a file on a specified disk on a specified system can simply request the file using some logical name, and a computer on the network which was designated to do the correlation then requests the file on the specified disk and system, treating the disk on that system as if it were the user's local disk. This translation of logical names to physical devices is transparent to the user. The computer doing the translating in this case is termed a disk or file server. Other server functions have been defined, such as a print server, which allows a file to be printed without specifying to which computer the printer is attached.

It is also possible to assign a user to a disk and then use the disk server to connect that user to that disk regardless of what computer on the LAN the user is on. A disk server program, termed a Local Area Disk (LAD), allows a user on one computer to treat a file on a disk on another computer as a virtual disk. This virtual disk acts as if it were a disk on the user's computer.

One of the files which can be accessed across the network are the operating system files. In fact, even before the LAD concept made the access to a file easy, many systems were developed to "downline" load an operating system to a computer on the network. It became unnecessary, therefore, that the computer, which is receiving the operating system, have an operating system on a disk or have a disk at all. For such a system to function, there are only two requirements. The first is that the network device be capable of generating a message and transmitting it over the communications link to request the operating system from another computer on the network. The second is that the network device be capable of loading the operating system it received into memory and causing the CPU to execute it.

In one prior arrangement, upon powering on the computer, the network device transmitted a request to be booted. This request, broadcast to all the other computers on the network, includes a simple message which contains the network device's hardware address. A computer on the network, upon receiving this request, checks its database to determine if it contains a listing identifying an operating system for the requesting computer. If the receiving computer finds the list entry of the requesting computer in its database, it becomes the host computer and retrieves the requested operating system from its disk, attaches the requesting computer's address to the file to form a message, and transmits the message on the network.

The requesting computer receives the message, copies it into its memory, and initiates the operating system execution. Although in principle the loading of an operating system into a computer on the network is simple to understand, its actual implemention is fairly complex.

To go into more detail, upon powering on, the requesting computer performs a self-test/power-up sequence. As part of the sequence, the processor looks for a boot device. If it fails to find a boot device, the processor will allow the network device to request a boot over the network.

Typically, the network device will not use the protocols usually used to communicate between computers. The reason for this is that such protocols are more complex than is necessary for such a simple task, and the network device would have to support a great deal of functionality it would not need. So instead, a simple protocol is used which consists of a small set of specialty messages for performing, testing, making boot requests, etc. (see for example DECnet^{T} Digital Network Architecture Maintenance Operations Functional Specification, order number (AA-x436A-TK DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS). The boot request message, for example, might simply contain the code number for a boot request, the code number for the type of device making the request, what type of computer the request is for, and whether the program requested is a boot-strap program or the operating system program. To these codes the network device of the requesting computer attaches its address, and an address indicating this is a broadcast to all other computers on the network.

When another computer on the network receives this message, it takes the hardware address of the message and looks through its list of computers for which it has operating system programs. If it fails to find the hardware address of the requesting computer listed in its database, the receiving computer simply ignores the message. If no computer has responded to the requesting computer within a certain amount of time, the requesting computer will again transmit a boot request. If the receiving computer finds the address of the requesting computer listed in its database, it then determines whether the requesting system's operating system is to be loaded immediately or in a series of steps.

The problem with this method of loading an operating system using the network is that too much information must be contained within the network prior to the boot request. That is, a computer node on the network needs to know, a priori, what programs, including boot-programs and operating system, each bootable network computer on the network would require. Further, for the operating system to work with the boot-programs, it is necessary that the operating system be modified to contain "hooks" or entry points that the boot-programs can call.

SUMMARY OF INVENTION

The invention as claimed in claim 1 provides a new and improved method of downloading operating systems or other executable programs to a computer on a network without a boot device and without requiring a modification of the loadable image, and as claimed in claim 11 a node for connection in a system.

Upon power up, the network device requests a boot and a minimum-boot program is transferred over the network into the requesting computer's memory. This minimum-boot program contains the functionality necessary to establish a network connection to a disk server on the network. When chained into the power-up/self-test program of the requesting computer, the minimum-boot program causes the computer to function as though it has a local disk with an operating system image, while actually transmitting all the disk accesses made by the computer across the network. Because the network is treated like a disk, there is no need to modify the operating system for downline loading.

BRIEF DESCRIPTION OF THE DRAWINGS

This invention is pointed out with particularity in the appended claims. The above and further advantages of this invention may be better understood by referring to the following description taken in conjunction with the accompanying drawings, in which:

Fig. 1 is block diagram of a system constructed in accordance with the invention including a host system and a node to be booted;

Fig. 2 is a block diagram of the minimum boot program structure;

Fig. 3 is a step diagram of the boot procedure.

DESCRIPTION OF AN ILLUSTRATIVE EMBODIMENT

Fig. 1 depicts a network consisting of a computer node 10, which does not include a local boot device, and a computer host node 14, which has a disk 20 containing the operating system of node 10, connected by a communications link 12. The node 10 has a communications interface, called the network device 26 by which it transmits and receives messages on the communications link 12. Similarly, node 14 has a communications interface 28.

The node 10 has many of the elements of an ordinary computer, such as a memory 27 and a central processing unit (CPU) 24. In addition it may have peripheral devices such as a video display terminal 16 and a local disk storage device 29.

The host 14 also has the elements of a computer including a CPU 30 and a memory 32 as well as peripherals such as a video display terminal 18 and a disk 20. The disk 20 on the host 14 not only contains the host's operating system 23 but also a minimum-boot program 34 needed to load the node 10 and a virtual disk file 33 containing the operating system 22 for node 10.

Upon powering up, the processor 24 of computer 10 executes a power power-on/self-test sequence during which it attempts to find a boot device which contains a file with its operating system in it. If it finds no boot device, the network device 26 broadcasts a request for an operating system over the network communications line 12. The network device 28 on the host 14 determines that the message is in broadcast mode and notifies the CPU 30. The CPU 30 of host 14 determines that the message is a boot request and searches its memory 32 to attempt to identify an operating system for the node 10. If it finds such an identification, it constructs a message by attaching the hardware address of the network device 26 of node computer 10 onto a minimum-boot program file 34 and places the message on the communications line 12.

The communications device 26 of the node 10 receives the message and copies the minimum-boot program into the node's low memory addresses 36 of its memory 27. The CPU 24 then begins to execute the minimum-boot program. The minimum-boot program is then copied into the high addresses 46 of memory 27 and the power-on/self-test sequence completes.

At this point the software on the node 10 is essentially treating the file 33 on disk 20 on host 14 as its virtual local disk. The node 10 reads a boot-block 56 from the virtual disk file 33 on disk 20 and uses it to load its operating system 22. It is important to note that all this is taking place over the network link 12 and not through access of the node's 10 local disk storage 29.

To understand the sequence of events in detail, it is first necessary to consider the nature of the minimum-boot program. The minimum-boot program, shown in Fig. 2, consists of a header 41 which identifies it as a network program to the host's operating system. The next portion is a boot-control program 42 which will prepare the node 10 for the downline load. A Local Area Disk (LAD) portion 48 is a program which allows a file on a disk on the network to be treated as a virtual local disk by node 10. A Local Area Systems Transport (LAST) 50 provides the actual local area network communications functions as described in co-pending application entitled "LOCAL AREA SYSTEM TRANSPORT" By Bruce E. Mann Et al. Attorney docket No. PD 88-0421. A Data Link Layer (DLL) 52 is the protocol providing the mechanism for communications between nodes through the network device 26 on node 10 and network device 28 on node 14. Finally, a Scheduler program 54 provides the real-time scheduling and timers required by network communications.

Fig. 3 shows in detail the steps performed by node 10 and host 14 during a downline load of the operating system for node 10. To the left of the figure are the steps performed by the node 10 and to the right are the steps performed by the host 14. Transfers between the two over the communications link 12 are shown in the center of the figure. The direction of the arrows indicates the direction of message flow.

With reference to Figs. 1, 2, and 3, when the computer 10 is powered on, it initiates a power-on/self-test (Step 100). During the power-on/self-test, the CPU 24 checks addresses on the bus to determine if an option board is present on the system. If it finds an option board, it processes any initialization routines located on the board. When the option board is a network device 26, the initialization routines (Step 102) on the device requests a boot from the network by broadcasting a boot request (Step 104). Another computer 14 on the network receives (Step 106) the boot request and searches (Step 108) its database for information concerning the requesting node 10. If the requesting node 10 is in the database, the appropriate minimum boot routine is recovered (Step 110) from disk 20 and transmitted (Step 112) to the requesting node 10.

Upon receipt (Step 114) of the minimum-boot routine by node 10, it copies it into low memory 36 and executes it (Step 116) in particular the boot control program 42. The minimum boot-control program 42 of the minimum-boot routine first determines (Step 118) the amount of memory 27 on node 10. In one embodiment, in which node 10 includes an IBM compatible personal computer, this is accomplished by initiating an interrupt-12 interrupt service routine. The boot-control 42 modifies (Step 120) the memory size reported by subtracting the size of the minimum boot program (Fig. 2) from the memory size returned by interrupt-12 in a location in memory. This is done to make this portion of memory unavailable to the node 10. The minimum boot routines are copied (Step 122) into high memory 46, and the boot-control is chained (Step 124) to a routine which enables a boot block to be read from a local disk. This is done to allow the power up/self-test sequence to complete and to regain control upon its completion.

The power-up/self-test sequence regains (Step 126) control upon completion of the network device 26 initialization routines and completes (Step 128). Upon completion, the power-up/self-test sequence attempts (Step 130) to read the boot block on the local disk. In this embodiment in which node 10 includes an IBM compatible personal computer, this is accomplished by initiating an interrupt-19 interrupt service routine. This causes the boot-control routine 42 to regain control (Step 132). The boot-control routine 42 calls (Step 134) the LAD program 48, the LAST program 50, the DLL program 52 and the Scheduler 54 at their entry points to initialize them. In this embodiment in which node 10 includes an IBM compatible personal computer, any disk access request generates an interrupt-13. LAD is chained to interrupt-13 so that any disk access request will be intercepted by LAD (Step 136).

The boot-control program 42 regains (Step 138) control, and enables the registers of the network device 26 to be read (Step 140) to determine the network address of node 10. The boot-control program 42 calls (Step 142) the LAD program 48 to establish a local area disk connection. Once the local area disk connection is established with server (Step 144), the virtual disk is available to the node. Once the local area disk connection has been established, the communications parameters may in fact, be changed. For example, in one embodiment, shown in Figs. 3C and 3D, it is at this time that the DECnet network parameters for the node 10, such as a network address, are read (Step 146) by the server so that the LAD routine 48 on the node 10 can be used to read (Step 148) the parameter file. Once the parameters are read, the LAD connection is broken (Step 150) and re-establshed (Step 152) using the new LAD parameters. In the general case where the communication parameters need not be changed, steps 146, 148, 150, and 152 need not be not taken and the node 10 continues with the boot procedure by requesting a read of boot-sector 56 to be retrieved.

The boot control program (42) issues a read request for the boot-sector 56 (Step 154). In one embodiment, in which node 10 includes an IBM compatible personal computer, this is accomplished by asserting interrupt-13. Since the LAD program 48 is chained in the interrupt service routine (Step 136) it intercepts this request and directs (Step 156) it, to the host 14. This host 14 reads (Step 158) the boot-sector 56 on the disk 20 and allows the node 10 to read it (Step 160). LAD 48 on the node 10 then exchanges messages (Step 162) with host 14 that facilitates the transfer of boot-sector 56 to node 10, and its execution (Step 164).

The boot-sector 56 enables the node 10 to request the operating system by asserting (Step 166) interrupt-13 which is again intercepted by LAD 48 and directed (Step 168) to the server. The server reads (Step 170) the operating system from file 33 on disk 20 and makes it available (Step 172) to LAD 48 on node 10. LAD 48 allows node 10 to read (Step 174) the operating system 22 and load it into memory 27. When all the operating system is loaded, LAD is unchained from interrupt-13 (Step 176) from LAD and returned to the local disk 29. Control is transferred to the initialization code of the executable program or operating system (step 178).

It should be stated that in this embodiment the host and the server were indicated to be the same node 14. This is not a requirement, and it is possible to have another implimentation in which they are separate.

It must be noted that none of the steps of the downline load of the operating system required that the operting system be modified in any way. By treating the host disk as if it were the local disk, any executable image may be loaded without modification.

Having shown the preferred embodiment, those skilled in the art will realize many variations are possible which will still be within the scope of the claimed invention. Therefore, it is the intention to limit the invention only as indicated by the scope of the claims.

Claims

1. A method of downline loading an executable image from a host computer (14) over a communications link (12) to a node (10), said node having an interface (26) through which it communicates over said communications link, comprising the steps of:

   a. enabling said node to retrieve (104-114) a remote minimum boot program (34) over said communications link and to link (124) said remote minimum-boot program (34) to a local image retrieval interrupt service routine;
   b. execution, (134) by said node, in response to a local image retrieval interrupt service request (130) generated by said node, of said remote minimum-boot program to initiate retrieval (144,158,170) of an executable image from said host;
   c. execution (178) by said node of said executable image;

   characterized in that said method further includes the step of:

   d. initiating execution (100,102) by said node of a node initialization program, said initialization program initiating execution of an interface initialization program to initialize said interface;
   e. execution, by said node of said interface initialization program to enable said node to retrieve said remote minimum-boot program, said

node thereafter completing execution of said node initialization program (102).

2. The method as defined in claim 1 in which said node (10), following retrieval of said executable image, thereafter removes (176) the remote minimum-boot program from said local image retrieval interrupt service routine.

3. The method of claim 1 wherein the execution of said interface initialization program comprises the steps of:

a. determining (102) that said interface (26) is configured to receive said remote minimum-boot program over said communications link (12); and
b. requesting (104) said remote minimum-boot program (34) over said communications link.

4. The method of claim 3 wherein said node (10) has a memory (27) having an address space which extends from a low address value to a high address value and in which said node, following the request of said remote minimum-boot program (34), thereafter :

a. loads (114) said remote minimum-boot program into said memory at a portion (36) of said address space proximate said low address space;
b. determines (118) the size of said memory on said node;
c. loads (120) in a memory size interrupt service location a new memory size value equal to the memory size minus the size of the remote minimum-boot program; and
d. copies (122) said remote minimum-boot program into said memory at a portion (46) of said address space proximate said high address value.

5. The method of claim 4 wherein the determining (118) of the amount of memory on said node comprises the steps of:

a. generating a request memory size interrupt service request; and
b. reading a value returned in said memory size interrupt service location.

6. The method of claim 1 wherein said generation (130) of said local image retrieval interrupt service request is in response to the completion (128) of the execution of said node initialization program.

7. The method of claim 1 wherein said execution (134) of said remote minimum-boot program comprises the steps of:

a. linking (136-142), by said node (10), of a local storage simulation program, which treats a storage device on the host computer as a local storage device, to a local storage device interrupt service routine;
b. establishing (144), by said node, of communication with said host; and
c. reading, (146) by said node, of said executable image located on a storage device on said host.

8. The method of claim 7 wherein the establishing (144) of communication with said host by said node comprises the steps of:

a. reading an address register of said interface to determine said address of said node;
b. generating a local storage device interrupt service request;
c. intercepting of said local storage device interrupt service request during processing of said local storage simulation program; and
d. using of said address during processing of said local storage simulation program to send a message to said host to establish a connection to said host storage device.

9. The method of claim 7 wherein said node, following said execution of said remote minimum-boot program, thereafter removes (150) said local storage simulation program from said local storage device interrupt service routine.

10. The method according to claim 1 wherein said execution of said interface initialization program includes the steps of:

i. issuing of a boot request by the node over said communications line;
ii. loading of said minimum-boot program into low memory of said node by said host;
iii. executing said minimum-boot by said node;
iv. copying of said minimum boot in said high memory of said node;
v. chaining said minimum-boot to an interrupt service routine of said node;
vi. completing a system selftest by said node

and wherein said execution of said remote minimum-boot program includes the steps of:

i. returning control of said minimum-boot by said system selftest;
ii. initializing a plurality of functions by said node;
iii. establishing a connection between said

node and said host;
iv. reading a boot sector on said host by said node;
v. loading said executable image;
vi. releasing LAD interrupt handling.

11. A node (10) for connection in a system including a host computer (14), a communications link (12) connecting said host to said node, and an interface (26) through which the node communicates over said communications link, said node comprising :

a. a means for enabling said node to retrieve a remote minimum-boot program (34) over said communications link (12) and to link said remote minimum-boot program to a local image retrieval interrupt service routine;
b. a means (24) for executing, following the linking of said remote minimum-boot program to said local image retrieval interrupt service routine and in response to a local image retrieval interrupt service request, said remote minimum-boot program to initiate retrieval of an executable image (22) from said host;
c. a means (24) for executing said executable image;

said node characterized by further including :

d. a means (24) for executing on said node a node initialization program including an interface initialization program;
e. a means (24) for executing said interface initialization program to enable operation of said means for enabling said mode to retrieve said remote minimum-boot program.

12. The node of claim 11, said node further comprising a means for removing said remote minimum-boot program from said local image retrieval interrupt service routine following the retrieval of said executable image.

13. The node of claim 11 wherein said processor for executing said interface initialization program comprises:

a. a means for determining said interface is configured to receive said remote minimum-boot program over said communications link, and
b. a means for requesting said remote minimum-boot program over said communications link.

14. The node of claim 12 further comprising :

a. a memory (27) having an address space which extends from a low address value to a high address value;
b. a means for loading said remote minimum-boot program into said memory at a portion (36) of said address space proximate said low address value following said request of said remote minimum boot program;
c. a means for determining the size of said memory;
d. a means for loading into a memory size interrupt service location a new memory size value equal to the memory size minus the size of the remote image retrieval program; and
e. a means for copying said remote minimum-boot program into memory of a portion (46) of said address space proximate said high address value.

15. The node in claim 13 wherein said means for determining the size of said memory comprises:

a. a means for generating a request memory size interrupt service request; and
b. a means for reading a value returned in said memory size interrupt service location.

16. The node in claim 11 further comprising a means for generating said local image retrieval interrupt service request in response to the completion of the execution of said node initialization program.

17. The node in claim 11 wherein said processor for executing said remote minimum-boot program comprises :

a. a means for linking a local storage simulation program which treats a storage device (20) on the host computer (14) as a local storage device to a local storage device interrupt service routine;
b. a means (26) for establishing communication with said host; and
c. a means for reading said executable image located on a storage device on said host.

18. The node of claim 16 wherein said means for establishing communications with said host comprises :

a. a means for reading an address register of said interface to determine said address of said node;
b. a means for generating a local storage device interrupt service request; and
c. a means for sending a message including said address of said node to said host to establish a connection to said host storage device in response to a local storage device interrupt

service request.

19. The node of claim 16 further comprising a means for removing said local storage simulation program from said local storage device interrupt service routine following said execution of said remote minimum-boot program.

Patentansprüche

1. Verfahren zum Herunterladen einer Abbildung eines ausführbaren Programms (executable image) von einem Host-Computer (14) über eine Kommunikationsverbindung (12) zu einem Knoten (10), wobei der Knoten eine Schnittstelle (26) aufweist, über die er über die Kommunikationsverbindung kommuniziert, wobei das Verfahren folgende Schritte aufweist:

a) den Knoten in die Lage zu versetzen, ein Fern-Minimum-Boot-Programm (34) über die Kommunikationsverbindung aufzufinden (104-114) und das Fern-Minimum-Boot-Programm (34) mit einer lokalen Abbildungs-Auffindungs-Interrupt-Dienstroutine zu verbinden;
b) Ausführen (134) des Fern-Minimum-Boot-Programms in Reaktion auf eine lokale Abbildungs-Auffindungs-Interrupt-Dienstaufforderung (130), die durch den Knoten erzeugt wird, um das Auffinden (144, 158, 170) einer Abbildung eines ausführbaren Programms von dem Host zu initiieren;
c) Ausführen (78) der Abbildung eines ausführbaren Programms durch den Knoten,

dadurch gekennzeichnet, daß das Verfahren ferner folgende Schritte umfaßt:

d) Initiieren der Ausführung (100, 102) eines Knoteninitialisierungsprogramms durch den Knoten, wobei das Initialisierungsprogramm die Ausführung eines Schnittstelleninitialisierungsprogrammes initiiert, um die Schnittstelle zu initialisieren;
e) Ausführen des Schnittstellen-Initialisierungsprogrammes durch den Knoten, um den Knoten in die Lage zu versetzen, das Fern-Minimum-Boot-Programm aufzufinden, woraufhin der Knoten die Ausführung des Knoten-Initialisierungsprogramms (102) abschließt.

2. Verfahren nach Anspruch 1, bei dem der Knoten (10) nach dem Auffinden der Abbildung des ausführbaren Programms das Fern-Minimum-Boot-Programm von der lokalen Abbildungs-Auffindungs-Interrupt-Dienstroutine entfernt (176).

3. Verfahren nach Anspruch 1, bei dem die Ausführung des Schnittstelleninitialisierungsprogramms folgende Schritte umfaßt:

a) Feststellen (102), daß die Schnittstelle (26) dazu konfiguriert ist, das Fern-Minimum-Boot-Programm über die Kommunikationsverbindung (12) zu empfangen; und
b) Anfordern (104) des Fern-Minimum-Boot-Programms (34) über die Kommunikationsverbindung.

4. Verfahren nach Anspruch 3, bei dem Knoten (10) einen Speicher (27) aufweist, der einen Adreßraum aufweist, der sich von einem niedrigen Adreßwert zu einem hohen Adreßwert erstreckt, und bei dem der Knoten nach der Anforderung des Fern-Minimum-Boot-Programms (34) folgendes durchführt:

a) Laden (114) des Fern-Minimum-Boot-Programms in den Speicher an einem Teil (36) des Adreßraums, der nahe an dem niedrigen Adreßwert liegt;
b) Ermitteln (118) der Größe des Speichers an dem Knoten;
c) Laden eines neuen Speichergrößenwerts, der gleich der Speichergröße minus der Größe des Fern-Minimum-Boot-Programms ist, an eine Speichergrößen-Interruptdienst-Speicherstelle; und
d) Kopieren (122) des Fern-Minimum-Boot-Programms in den Speicher an einen Teil (46) des Adreßraums, der nahe bei dem hohen Adreßwert liegt.

5. Verfahren nach Anspruch 4, bei dem das Bestimmen der Speichergröße an dem Knoten folgende Schritte umfaßt:

a) Erzeugen einer Speichergrößen-Interrupt-Dienstanforderung; und
b) Lesen eines an der Speichergrößen-Interruptdienst-Speicherstelle zurückgelieferten Wertes.

6. Verfahren nach Anspruch 1, bei dem das Erzeugen (130) der lokalen Abbildungs-Auffindungs-Interrupt-Dienstanforderung in Reaktion auf die vollständige Ausführung (128) des Knoten-Initialisierungsprogramms geschieht.

7. Verfahren nach Anspruch 1, bei dem die Ausführung (134) des Fern-Minimum-Boot-Programms folgende Schritte umfaßt:

a) Binden (136-142) eines lokalen Speicher-Simulationsprogramms, welches eine Speichereinrichtung auf dem Host-Computer als eine lo-

kale Speichereinrichtung behandelt, durch den Knoten (10) mit einer lokalen Speichereinrichtungs-Interrupt-Dienstroutine;
b) Herstellen (144) der Kommunikation mit dem Host durch den Knoten; und
c) Lesen (146) der in einer Speichereinrichtung auf dem Host befindlichen Abbildung des ausführbaren Programms durch den Knoten.

8. Verfahren nach Anspruch 7, bei dem die Herstellung (144) der Kommunikation mit dem Host durch den Knoten folgende Schritte umfaßt:

a) Lesen eines Adreßregisters der Schnittstelle, um die Adresse des Knotens zu bestimmen;
b) Erzeugen einer lokalen Speichereinrichtungs-Interrupt-Dienstanforderung;
c) Abfangen der lokalen Speichereinrichtungs-Interrupt-Dienstanforderung während der Verarbeitung des lokalen Speicher-Simulationsprogramms; und
d) Verwenden der Adresse während der Verarbeitung des lokalen Speicher-Simulationsprogramms, um eine Nachricht an den Host zu senden, um eine Verbindung mit der Host-Speichereinrichtung herzustellen.

9. Verfahren nach Anspruch 7, bei dem der Knoten nach der Ausführung des Fern-Minimum-Boot-Programms das lokalen Speicher-Simulationsprogramm von der lokalen Speichereinrichtungs-Interrupt-Dienstroutine entfernt (150).

10. Verfahren nach Anspruch 1, bei dem die Ausführung des Schnittstelleninitialisierungsprogramms folgende Schritte umfaßt:

i) Ausgeben einer Boot-Anforderung durch den Knoten über die Kommunikationsleitung;
ii) Laden des Fern-Minimum-Boot-Programms in einen niedrigen Speicherbereich des Knotens durch den Host;
iii) Ausführung des Fern-Minimum-Boot-Programms durch den Knoten;
iv) Kopieren des Fern-Minimum-Boot-Programms in den hohen Speicherbereich des Knotens;
v) Verbinden des Fern-Minimum-Boot-Programms mit einer Interrupt-Dienstroutine des Knotens;
vi) Ausführen eines System-Selbsttests durch den Knoten;

wobei die Ausführung des Fern-Minimum-Boot-Programms folgende Schritte umfaßt:

i) Zurückgabe der Kontrolle des Fern-Minimum-Boot-Programms durch den System-Selbsttest;
ii) Initialisieren einer Vielzahl von Funktionen durch den Knoten;
iii) Herstellen einer Verbindung zwischen dem Knoten und dem Host;
iv) Lesen eines Boot-Sektors auf dem Host durch den Knoten;
v) Laden der Abbildung des ausführbaren Programms;
vi) Freigeben der LAD(Local Area Disk)-Interruptbearbeitung.

11. Knoten (10) zum Anschluß in einem System, welches einen Host-Computer (14), eine Kommunikationsverbindung (12), die den Host mit dem Knoten verbindet, sowie eine Schnittstelle (26), über die der Knoten über die Kommunikationsverbindung kommuniziert, einschließt, wobei der Knoten umfaßt:

a) eine Einrichtung, die den Knoten in die Lage versetzt, ein Fern-Minimum-Boot-Programm (34) über die Kommunikationsverbindung (12) aufzufinden und das Fern-Minimum-Boot-Programm mit einer lokalen Abbildungs-Auffindungs-Interrupt-Dienstroutine zu binden;
b) eine Einrichtung (24), um nach dem Binden des Fern-Minimum-Boot-Programms mit der lokalen Abbildungs-Auffindungs-Interrupt-Dienstroutine und in Reaktion auf eine lokale Abbildungs-Auffindungs-Interrupt-Dienstanforderung das Fern-Minimum-Boot-Programm auszuführen, um das Auffinden einer Abbildung des ausführbaren Programms (22) von dem Host zu starten;
c) eine Einrichtung (24), um die Abbildung des ausführbaren Programms auszuführen, wobei der Knoten dadurch gekennzeichnet ist, daß er ferner einschließt:
d) eine Einrichtung (24), um auf dem Knoten ein Knoten-Initialisierungsprogramm auszuführen, welches ein Schnittstelleninitialisierungsprogramm umfaßt;
e) eine Einrichtung (24) zum Ausführen des Schnittstellen-Initialisierungsprogramms, um den Betrieb der Einrichtung zu ermöglichen, die es dem Knoten ermöglicht, das Fern-Minimum-Boot-Programm abzurufen.

12. Knoten nach Anspruch 11, wobei der Knoten ferner eine Einrichtung zum Entfernen des Fern-Minimum-Boot-Programms von der lokalen Abbildungs-Auffindungs-Interrupt-Dienstroutine nach dem Auffinden des ausführbaren Programms umfaßt.

13. Knoten nach Anspruch 11, bei dem der Prozessor zum Ausführen des Schnittstelleninitialisierungsprogramms folgendes umfaßt:

a) eine Einrichtung, um festzustellen, daß die Schnittstelle konfiguriert ist, das Fern-Minimum-Boot-Programm über die Kommunikationsverbindung zu empfangen; und
b) eine Einrichtung zum Anfordern des Fern-Minimum-Boot-Programms über die Kommunikationsverbindung.

14. Knoten nach Anspruch 12, welcher ferner umfaßt:

a) einen Speicher (27), der einen Adreßraum aufweist, der sich von einem niedrigen Adreßwert zu einem hohen Adreßwert erstreckt.
b) eine Einrichtung zum Laden des Fern-Minimum-Boot-Programms in den Speicher in einem Bereich (36) des Adreßraums, der nahe dem niedrigen Adreßwert liegt, und zwar nach der Anforderung des Fern-Minimum-Boot-Programms;
c) eine Einrichtung zum Bestimmen der Größe des Speichers;
d) eine Einrichtung zum Laden eines neuen Speichergrößenwerts, der gleich der Speichergröße minus der Größe des Fern-Minimum-Boot-Programms ist, an eine Speichergrößen-Interruptdienst-Speicherstelle; und
e) eine Einrichtung zum Kopieren des Fern-Minimum-Boot-Programms in einen Speicherbereich (46) des Adreßraums, der nahe dem hohen Adreßwert liegt.

15. Knoten nach Anspruch 13, bei dem die Einrichtung zum Bestimmen der Speichergröße folgendes umfaßt:

a) eine Einrichtung zum Erzeugen einer Speichergrößen-Interrupt-Dienstanforderung; und
b) eine Einrichtung zum Lesen eines an der Speichergrößen-Interruptdienst-Speicherstelle zurückgegebenen Wertes.

16. Knoten nach Anspruch 11, welcher ferner eine Einrichtung zum Erzeugen der lokalen Abbildungs-Auffindungs-Interrupt-Dienstanforderung in Reaktion auf die vollständige Ausführung des Knoten-Initialisierungsprogramms umfaßt.

17. Knoten nach Anspruch 11, bei dem der Prozessor zum Ausführen des Fern-Minimum-Boot-Programms umfaßt:

a) eine Einrichtung zum Binden eines lokalen Speicher-Simulationsprogramms, welches eine Speichereinrichtung (20) auf dem Host-Computer (14) als eine lokale Speichereinrichtung behandelt, mit einer lokalen Speichereinrichtungs-Interrupt-Dienstroutine;
b) eine Einrichtung (26) zum Herstellen der Kommunikation mit dem Host; und
c) eine Einrichtung zum Lesen der Abbildung des ausführbaren Programms, die sich auf einer Speichereinrichtung auf dem Host befindet.

18. Knoten nach Anspruch 16, bei dem die Einrichtung zum Herstellen der Kommunikation mit dem Host folgendes umfaßt:

a) eine Einrichtung zum Lesen eines Adreßregisters der Schnittstelle, um die Adresse des Knotens zu bestimmen;
b) eine Einrichtung zum Erzeugen einer lokalen Speichereinrichtungs-Interrupt-Dienstanforderung; und
c) Einrichtung zum Senden einer Nachricht, welche die Adresse des Knotens einschließt, an den Host, um eine Verbindung mit der Host-Speichereinrichtung herzustellen, in Reaktion auf eine lokalen Speichereinrichtungs-Interrupt-Dienstanforderung.

19. Knoten nach Anspruch 16, welcher ferner eine Einrichtung zum Entfernen des lokalen Speicher-Simulationsprogramms von der lokalen Speichereinrichtungs-Interrupt-Dienstroutine nach der Ausführung des Fern-Minimum-Boot-Programms umfaßt.

Revendications

1. Méthode de téléchargement d'une image exécutable à partir d'un ordinateur hôte (14) par l'intermédiaire d'une liaison de communications (12) vers un noeud (10), ledit noeud comportant une interface (26) grâce à laquelle il communique par l'intermédiaire de ladite liaison de communications, comprenant les étapes :

a. d'activation dudit noeud pour rechercher (104-114) un programme de chargement minimum à distance (34) par l'intermédiaire de ladite liaison de communications et pour associer (124) ledit programme de chargement minimum à distance (34) à un sous-programme de prise en charge d'interruption pour recherche d'image locale;
b. d'exécution (134) par ledit noeud, en réponse à une demande de prise en charge d'interruption pour recherche d'image locale (130) émise par ledit noeud, dudit programme de chargement minimum à distance pour déclencher la recherche (144, 158, 170) d'une image exécutable à partir dudit hôte;
c. d'exécution (178) par ledit noeud de ladite image exécutable;

caractérisée en ce que ladite méthode comprend également l'étape :

d. de lancement de l'exécution (100, 102) par ledit noeud d'un programme d'initialisation de noeud, ledit programme d'initialisation lançant l'exécution d'un programme d'initialisation d'interface pour initialiser ladite interface;
e. d'exécution par ledit noeud dudit programme d'initialisation d'interface pour permettre audit noeud de rechercher ledit programme de chargement minimum à distance, ledit noeud achevant ensuite l'exécution (102) dudit programme d'initialisation de noeud.

2. Méthode telle que définie dans la revendication 1, selon laquelle, après la recherche de ladite image exécutable, ledit noeud (10) retire (176) le programme de chargement minimum à distance dudit sous-programme de prise en charge d'interruption pour recherche d'image locale.

3. Méthode selon la revendication 1, suivant laquelle l'exécution dudit programme d'initialisation d'interface comprend les étapes de :

a. détermination (102) que ladite interface (26) est configurée pour recevoir ledit programme de chargement minimum à distance par l'intermédiaire de ladite liaison de communications (12); et
b. demande (104) dudit programme de chargement minimum à distance (34) par l'intermédiaire de ladite liaison de communications.

4. Méthode selon la revendication 3, suivant laquelle ledit noeud (10) possède une mémoire (27) comportant un espace d'adresses qui s'étend d'une valeur d'adresse basse à une valeur d'adresse haute, et suivant laquelle, après la demande dudit programme de chargement minimum à distance (34), ledit noeud :

a. entre (114) ledit programme de chargement minimum à distance dans ladite mémoire au niveau d'une partie (36) dudit espace d'adresses, proche de ladite valeur d'adresse basse:
b. détermine (118) la capacité de sa mémoire;
c. entre (120) dans un emplacement de prise en charge d'interruption pour capacité de mémoire une nouvelle valeur de capacité de mémoire égale à la capacité de la mémoire moins le volume du programme de chargement minimum à distance; et
d. copie (122) ledit programme de chargement minimum à distance dans ladite mémoire au niveau d'une partie (46) dudit espace d'adresses, proche de ladite valeur d'adresse haute.

5. Méthode selon la revendication 4, suivant laquelle la détermination (118) de la capacité de mémoire dudit noeud comprend les étapes :

a. d'émission d'une demande de prise en charge d'interruption pour capacité de mémoire; et
b. de lecture d'une valeur renvoyée dans ledit emplacement de prise en charge d'interruption pour capacité de mémoire.

6. Méthode selon la revendication 1, suivant laquelle ladite émission (130) de ladite demande de prise en charge d'interruption pour recherche d'image locale a lieu en réponse à l'achèvement (128) de l'exécution dudit programme d'initialisation de noeud.

7. Méthode selon la revendication 1, suivant laquelle ladite exécution (134) dudit programme de chargement minimum à distance comprend les étapes :

- a. d'association (136-142) par ledit noeud (10) d'un programme de simulation de stockage local qui traite un dispositif de stockage de l'ordinateur hôte comme un dispositif de stockage local, à un sous-programme de prise en charge d'interruption pour dispositif de stockage local;
b. d'établissement (144) par ledit noeud d'une communication avec ledit hôte; et
c. de lecture (146) par ledit noeud de ladite image exécutable située dans un dispositif de stockage dudit hôte.

8. Méthode selon la revendication 7, suivant laquelle l'établissement (144) par ledit noeud d'une communication avec ledit hôte comprend les étapes :

a. de lecture d'un registre d'adresse de ladite interface pour déterminer ladite adresse dudit noeud;
b. d'émission d'une demande de prise en charge d'interruption pour dispositif de stockage local;
c. d'interception de ladite demande de prise en charge d'interruption pour dispositif de stockage local au cours du traitement dudit programme de simulation de stockage local; et
d. d'utilisation de ladite adresse au cours du traitement dudit programme de simulation de stockage local pour envoyer un message audit hôte afin d'établir une connexion avec son dispositif de stockage local.

9. Méthode selon la revendication 7, suivant laquelle, après ladite exécution dudit programme de chargement minimum à distance, ledit noeud retire (150) ledit programme de simulation de stockage local dudit sous-programme de prise en charge d'interruption pour dispositif de stockage local.

10. Méthode selon la revendication 1, suivant laquelle ladite exécution dudit programme d'initialisation d'interface comprend les étapes :

i. d'émission d'une demande de chargement par le noeud par l'intermédiaire de ladite ligne de communication;
ii. d'entrée dudit programme de chargement minimum dans une partie basse de la mémoire dudit noeud par ledit hôte;
iii. d'exécution dudit chargement minimum par ledit noeud;
iv. de copie dudit chargement minimum dans la partie haute de la mémoire dudit noeud;
v. d'enchaînement dudit chargement minimum avec un sous-programme de prise en charge d'interruption dudit noeud;
vi. d'exécution par ledit noeud d'un test automatique de système

et suivant laquelle ladite exécution dudit programme de chargement minimum à distance comprend les étapes :

i. de renvoi du contrôle dudit chargement minimum par ledit test automatique de système;
ii. d'initialisation de plusieurs fonctions par ledit noeud;
iii. d'établissement d'une connexion entre ledit noeud et ledit hôte;
iv. de lecture d'un secteur de chargement sur ledit hôte par ledit noeud;
v. d'entrée de ladite image exécutable;
vi. de fin de prise en charge d'interruption LAD.

11. Noeud (10) destiné à être connecté dans un système comprenant un ordinateur hôte (14), une liaison de communications (12) reliant ledit hôte audit noeud, et une interface (26) grâce à laquelle le noeud communique par l'intermédiaire de ladite liaison de communications, ledit noeud comprenant :

a. un moyen destiné à permettre audit noeud de rechercher un programme de chargement minimum à distance (34) par l'intermédiaire de ladite liaison de communication (12) et d'associer ledit programme de chargement minimum à distance à un sous-programme de prise en charge d'interruption pour recherche d'image locale;
b. un moyen (24) destiné à exécuter, après l'association dudit programme de chargement minimum à distance avec ledit sous-programme de prise en charge d'interruption pour recherche d'image locale et en réponse à une demande de prise en charge d'interruption pour recherche d'image locale, ledit programme de chargement minimum à distance afin de déclencher la recherche d'une image exécutable (22) à partir dudit hôte;
c. un moyen (24) destiné à exécuter ladite image exécutable;

ledit noeud étant caractérisé en ce qu'il comprend également :

d. un moyen (24) destiné à exécuter sur ledit noeud un programme d'initialisation de noeud comprenant un programme d'initialisation d'interface;
e. un moyen (24) destiné à exécuter ledit programme d'initialisation d'interface pour permettre le fonctionnement dudit moyen destiné à permettre audit noeud de rechercher ledit programme de chargement minimum à distance.

12. Noeud selon la revendication 11, ledit noeud comprenant également un moyen destiné à dissocier ledit programme de chargement minimum à distance dudit sous-programme de prise en charge d'interruption pour recherche d'image locale, après la recherche de ladite image exécutable.

13. Noeud selon la revendication 11, dans lequel ledit processeur destiné à exécuter ledit programme d'initialisation d'interface comprend :

a. un moyen destiné à déterminer que ladite interface est configurée pour recevoir ledit programme de chargement minimum à distance par l'intermédiaire de ladite liaison de communications, et
b. un moyen destiné à demander ledit programme de chargement minimum à distance par l'intermédiaire de ladite liaison de communications.

14. Noeud selon la revendication 12, comprenant également :

a. une mémoire (27) comportant un espace d'adresses qui s'étend d'une valeur d'adresse basse à une valeur d'adresse haute;
b. un moyen destiné à entrer ledit programme de chargement minimum à distance dans ladite mémoire au niveau d'une partie (36) dudit espace d'adresses, proche de ladite valeur d'adresse basse après la demande dudit programme de chargement minimum à distance;
c. un moyen destiné à déterminer la capacité de ladite mémoire;
d. un moyen destiné à entrer dans un emplacement de prise en charge d'interruption pour capacité de mémoire une nouvelle valeur de capacité de mémoire égale à la capacité du la

mémoire moins le volume du programme de recherche d'image à distance; et
e. un moyen destiné à copier ledit programme de chargement minimum à distance dans la mémoire au niveau d'une partie (46) dudit espace d'adresses, proche de ladite valeur d'adresse haute.

15. Noeud selon la revendication 13, dans lequel ledit moyen destiné à déterminer la capacité de ladite mémoire comprend :

a. un moyen destiné à émettre une demande de prise en charge d'interruption pour capacité de mémoire; et
b. un moyen destiné à lire une valeur renvoyée dans ledit emplacement de prise en charge d'interruption pour capacité de mémoire.

16. Noeud selon la revendication 11, comprenant également un moyen pour émettre ladite demande de prise en charge d'interruption pour recherche d'image locale en réponse à l'achèvement de l'exécution dudit programme d'initialisation de noeud.

17. Noeud selon la revendication 11, dans lequel ledit processeur destiné à exécuter ledit programme de chargement minimum à distance comprend :

a. un moyen destiné à associer un programme de simulation de stockage local qui traite un dispositif de stockage (20) de l'ordinateur hôte (14) comme un dispositif de stockage local à un sous-programme de prise en charge d'interruption pour dispositif de stockage local;
b. un moyen (26) destiné à établir une communication avec ledit hôte; et
c. un moyen destiné à lire ladite image exécutable située dans un dispositif de stockage dudit hôte.

18. Noeud selon la revendication 16, dans lequel ledit moyen destiné à établir des communications avec ledit hôte comprend :

a. un moyen destiné à lire un registre d'adresse de ladite interface pour déterminer ladite adresse dudit noeud;
b. un moyen destiné à émettre une demande de prise en charge d'interruption pour dispositif de stockage local; et
c. un moyen destiné à envoyer un message contenant ladite adresse dudit noeud audit hôte pour établir une connexion avec ledit dispositif de stockage de l'hôte en réponse à une demande de prise en charge d'interruption pour dispositif de stockage local.

19. Noeud selon la revendication 16, comprenant également un moyen destiné à dissocier ledit programme de simulation de stockage local dudit sous-programme de prise en charge d'interruption pour dispositif ce stockage local après ladite exécution dudit programme de chargement minimum à distance.

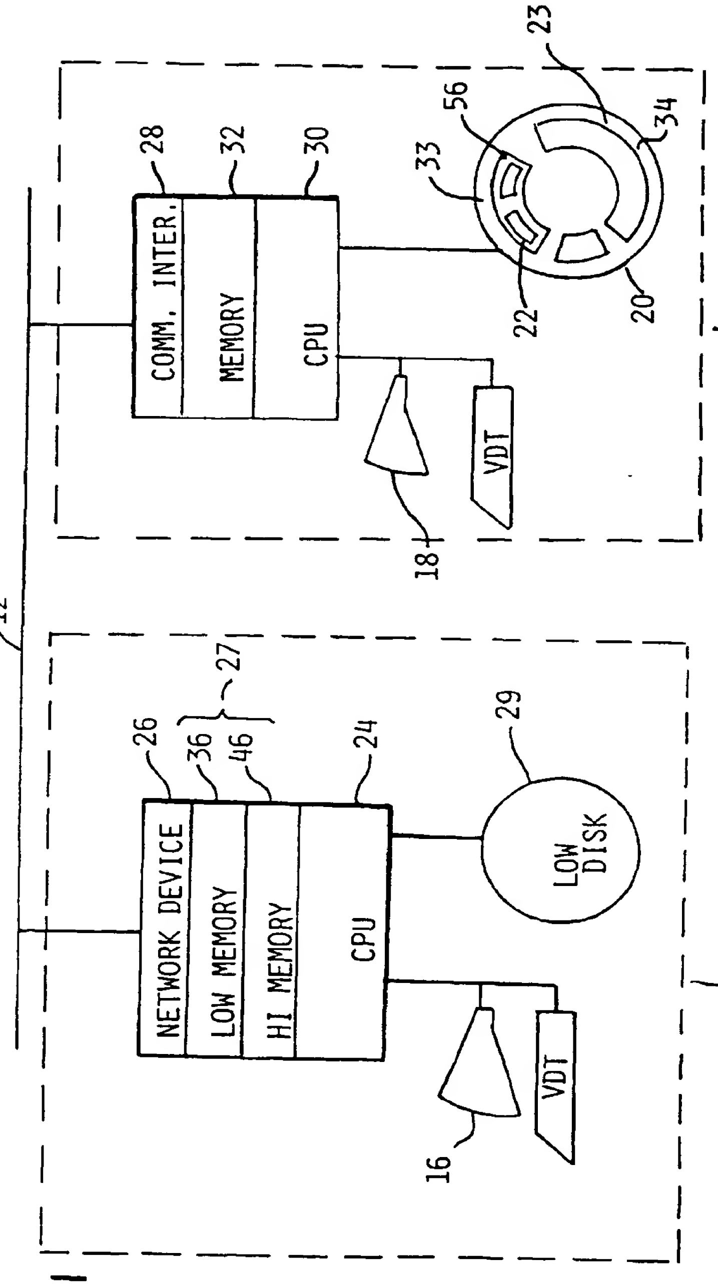
COMM. INTER.
MEMORY
CPU
VDT
28
32
30
56
33
23
34
22
20
14
18
12
NETWORK DEVICE
LOW MEMORY
HI MEMORY
CPU
LOW DISK
VDT
26
36
46
27
24
29
16
10

FIG. 1

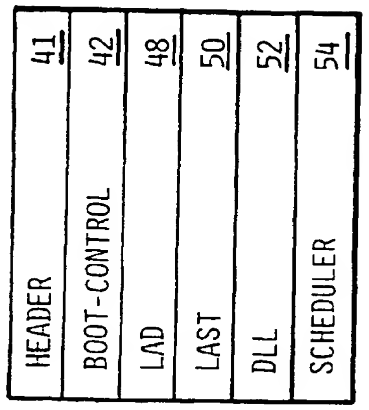
HEADER 41
BOOT-CONTROL 42
LAD 48
LAST 50
DLL 52
SCHEDULER 54

FIG. 2

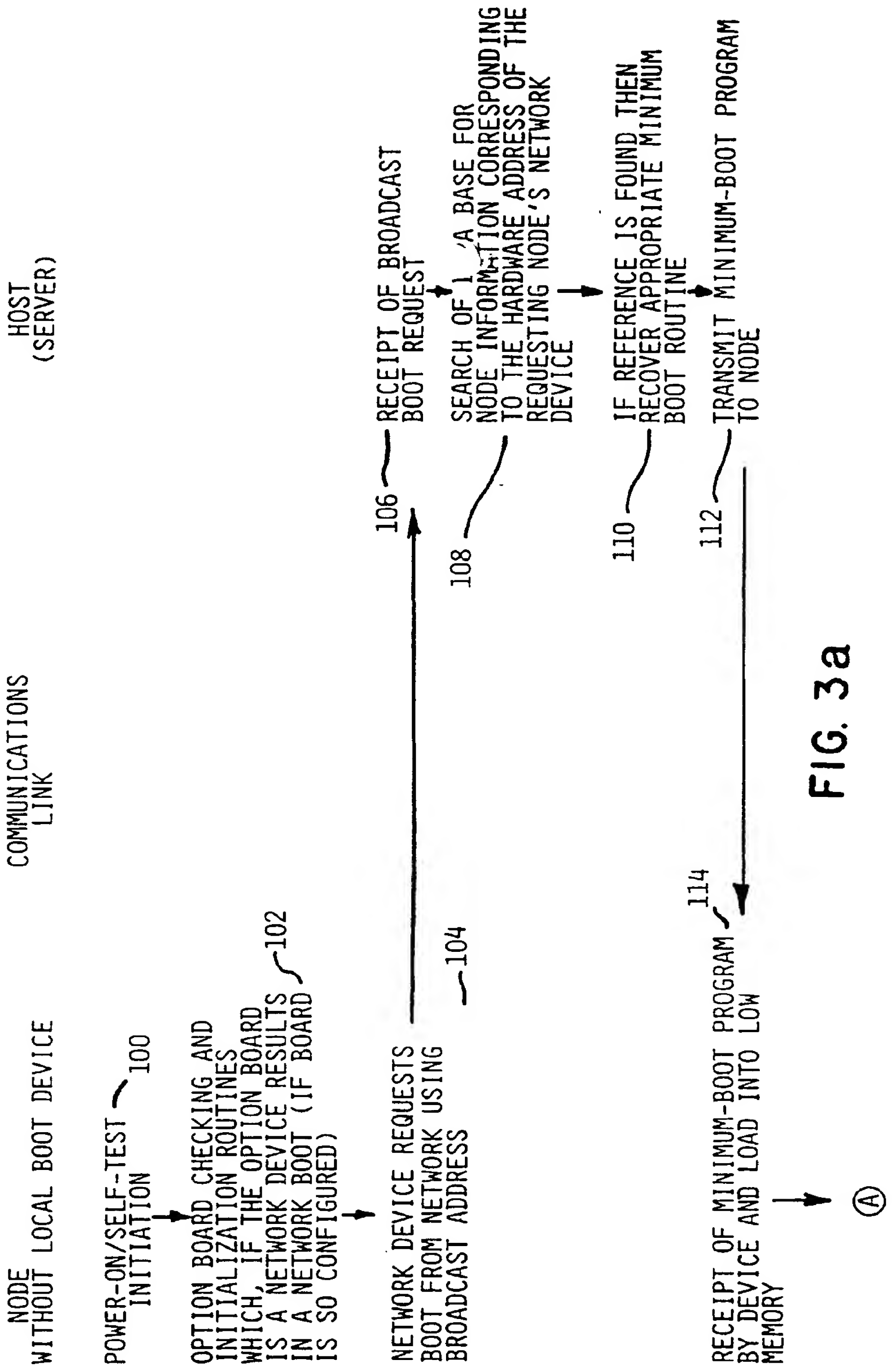
NODE WITHOUT LOCAL BOOT DEVICE
COMMUNICATIONS LINK
HOST (SERVER)
POWER-ON/SELF-TEST INITIATION
100
OPTION BOARD CHECKING AND INITIALIZATION ROUTINES WHICH, IF THE OPTION BOARD IS A NETWORK DEVICE RESULTS IN A NETWORK BOOT (IF BOARD IS SO CONFIGURED)
102
NETWORK DEVICE REQUESTS BOOT FROM NETWORK USING BROADCAST ADDRESS
104
RECEIPT OF BROADCAST BOOT REQUEST
106
SEARCH OF DATA BASE FOR NODE INFORMATION CORRESPONDING TO THE HARDWARE ADDRESS OF THE REQUESTING NODE'S NETWORK DEVICE
108
IF REFERENCE IS FOUND THEN RECOVER APPROPRIATE MINIMUM BOOT ROUTINE
110
TRANSMIT MINIMUM-BOOT PROGRAM TO NODE
112
RECEIPT OF MINIMUM-BOOT PROGRAM BY DEVICE AND LOAD INTO LOW MEMORY
114
A

FIG. 3a

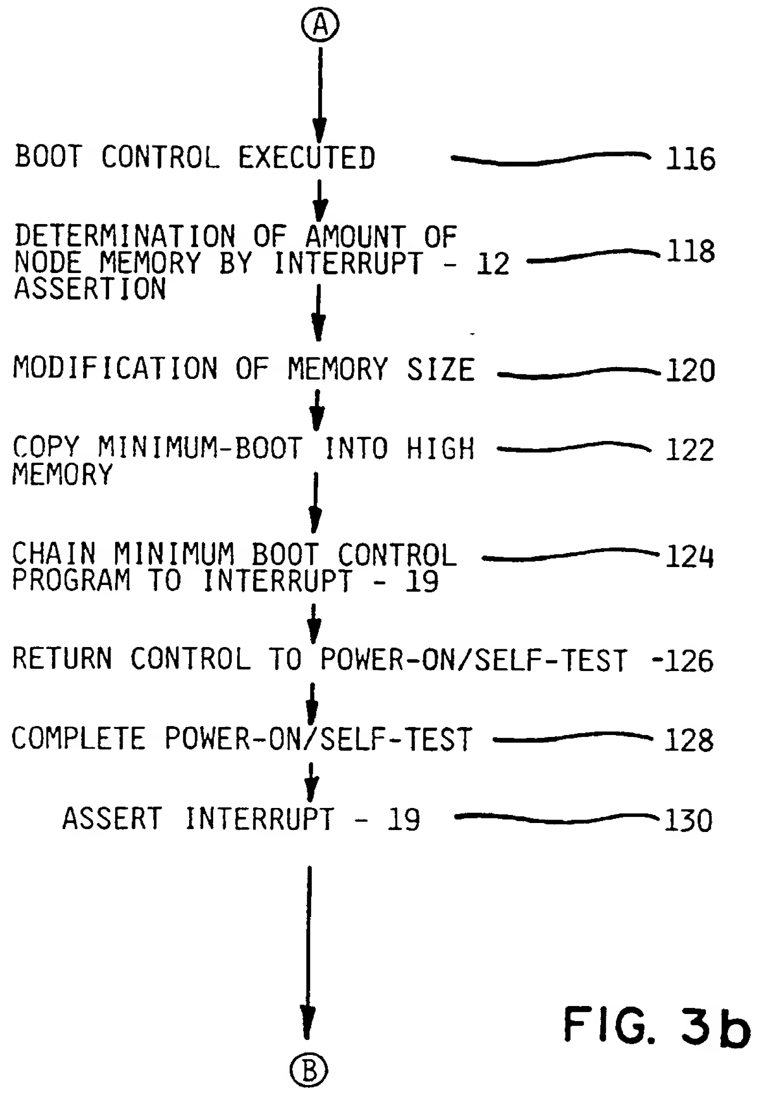
A
BOOT CONTROL EXECUTED
116
DETERMINATION OF AMOUNT OF NODE MEMORY BY INTERRUPT - 12 ASSERTION
118
MODIFICATION OF MEMORY SIZE
120
COPY MINIMUM-BOOT INTO HIGH MEMORY
122
CHAIN MINIMUM BOOT CONTROL PROGRAM TO INTERRUPT - 19
124
RETURN CONTROL TO POWER-ON/SELF-TEST
126
COMPLETE POWER-ON/SELF-TEST
128
ASSERT INTERRUPT - 19
130
B

FIG. 3b

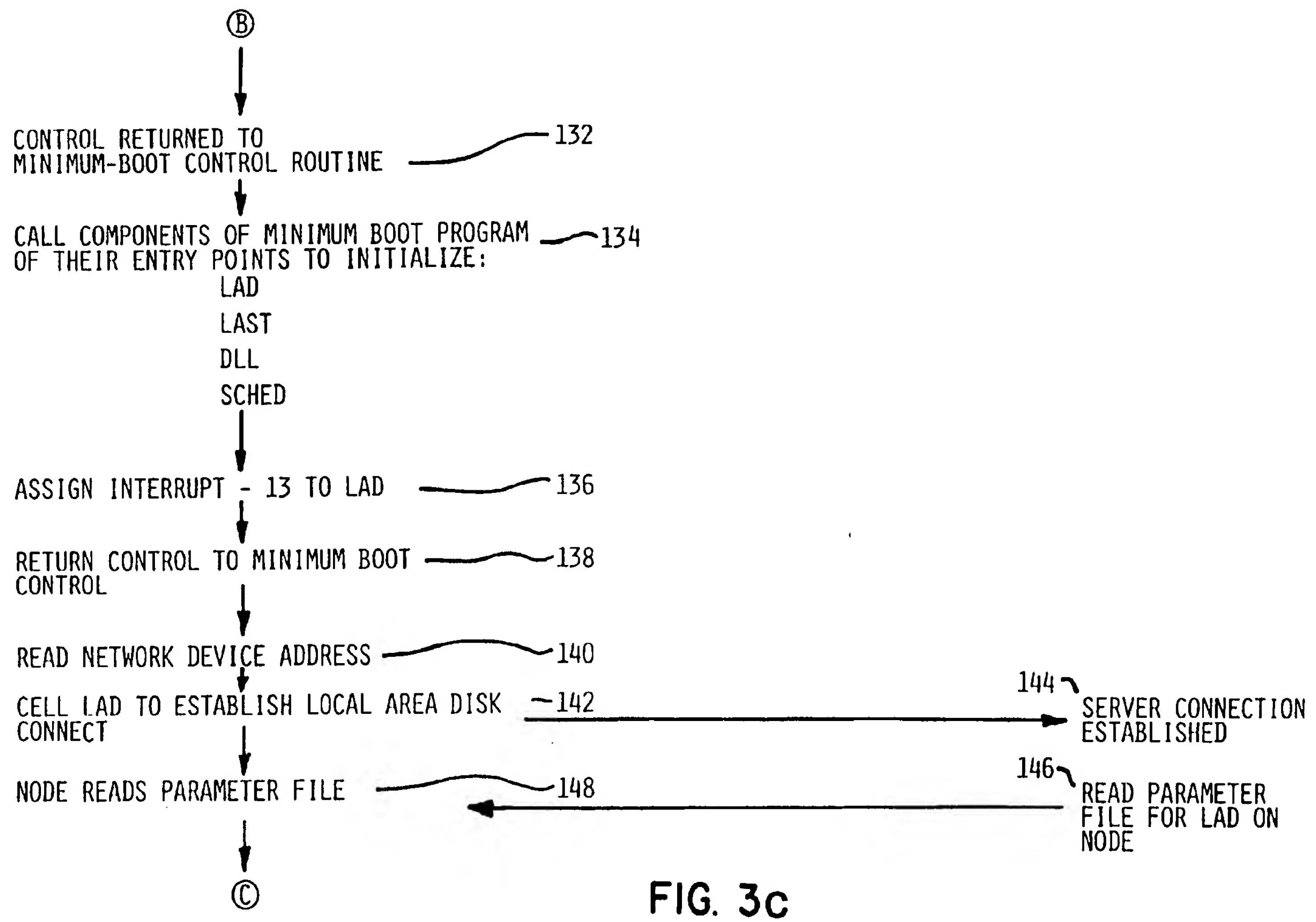
B
CONTROL RETURNED TO MINIMUM-BOOT CONTROL ROUTINE
132
CALL COMPONENTS OF MINIMUM BOOT PROGRAM OF THEIR ENTRY POINTS TO INITIALIZE:
LAD
LAST
DLL
SCHED
134
ASSIGN INTERRUPT - 13 TO LAD
136
RETURN CONTROL TO MINIMUM BOOT CONTROL
138
READ NETWORK DEVICE ADDRESS
140
CELL LAD TO ESTABLISH LOCAL AREA DISK CONNECT
142
144
SERVER CONNECTION ESTABLISHED
146
READ PARAMETER FILE FOR LAD ON NODE
NODE READS PARAMETER FILE
148
C

FIG. 3c

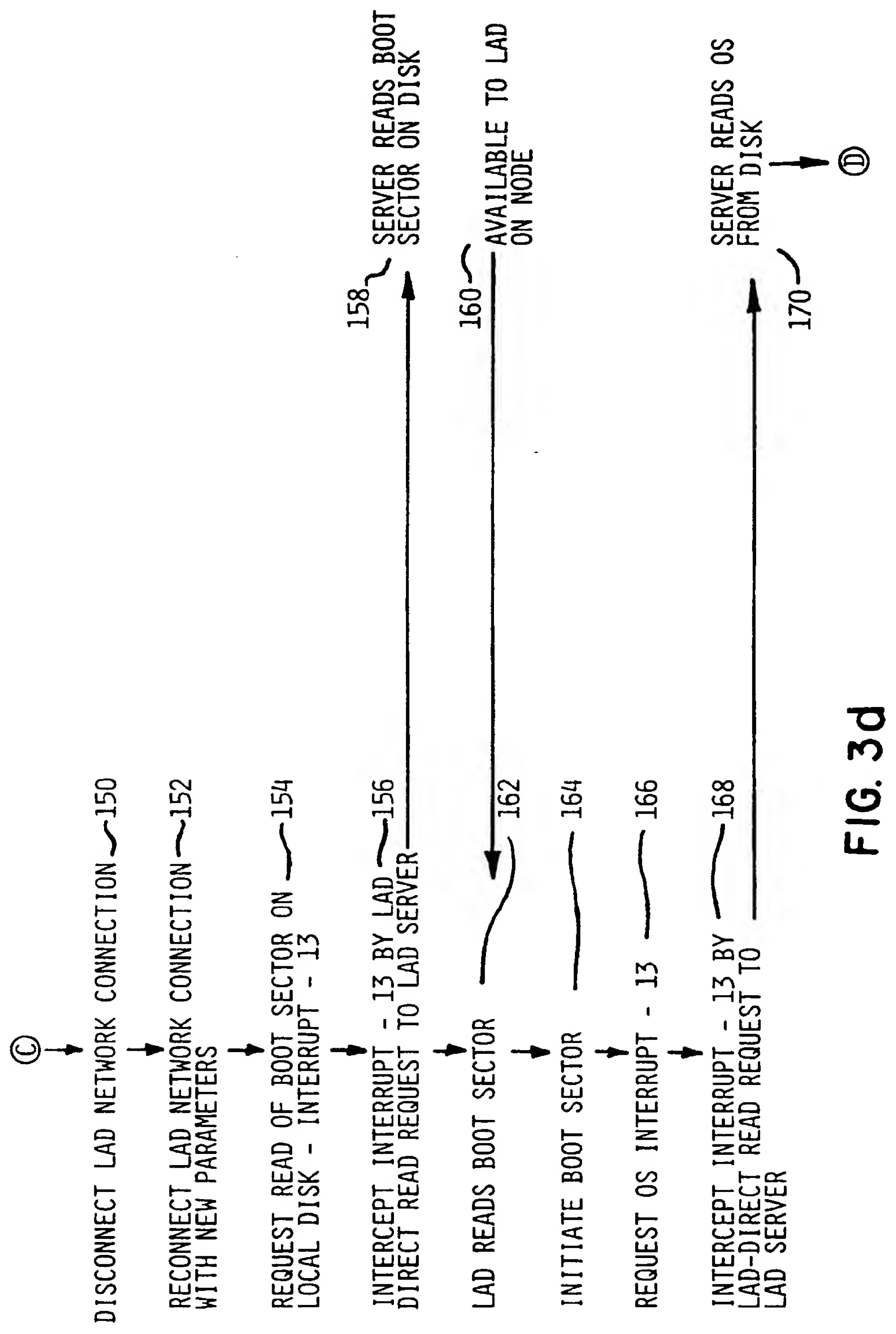
C
DISCONNECT LAD NETWORK CONNECTION
150
RECONNECT LAD NETWORK CONNECTION WITH NEW PARAMETERS
152
REQUEST READ OF BOOT SECTOR ON LOCAL DISK - INTERRUPT - 13
154
INTERCEPT INTERRUPT - 13 BY LAD DIRECT READ REQUEST TO LAD SERVER
156
158
SERVER READS BOOT SECTOR ON DISK
160
AVAILABLE TO LAD ON NODE
LAD READS BOOT SECTOR
162
INITIATE BOOT SECTOR
164
REQUEST OS INTERRUPT - 13
166
INTERCEPT INTERRUPT - 13 BY LAD-DIRECT READ REQUEST TO LAD SERVER
168
170
SERVER READS OS FROM DISK
D

FIG. 3d

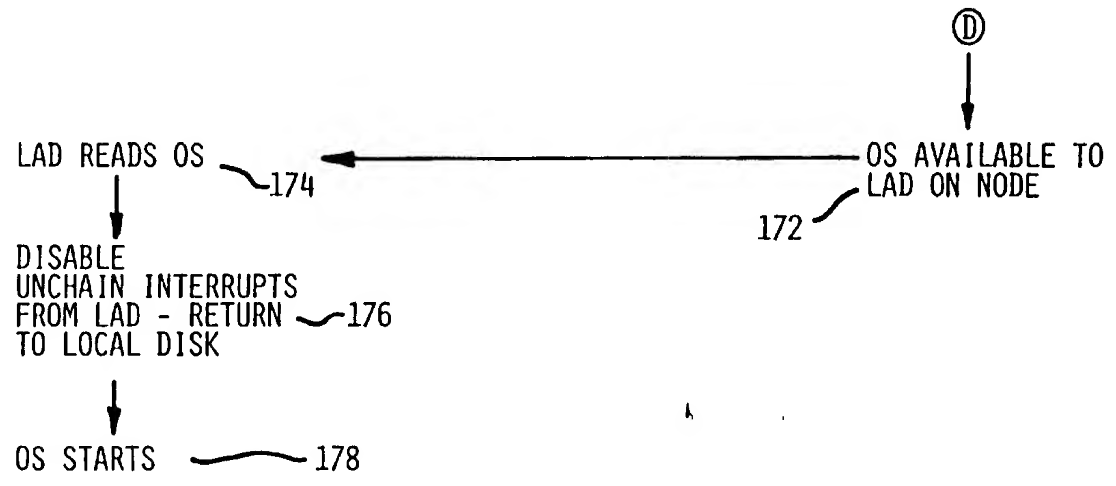
D
OS AVAILABLE TO LAD ON NODE
172
LAD READS OS
174
DISABLE UNCHAIN INTERRUPTS FROM LAD - RETURN TO LOCAL DISK
176
OS STARTS
178

FIG. 3e